Aufgabenblatt Nr. 6 inf11 22.03.2021
mit Lösungen zu Nr. 15 – Nr. 18 aus der BBB-Konferenz vom 25.03.2021

15. **BOOLESCHE TERME**

Schreibweisen für die Negation:
not a = ¬ a = $\overline{a}$

Wir ergänzen die in Aufgabe 11 (Blatt 4, 26.01.2021) für Boolesche Variable formulierten Rechenregeln
- Kommutativgesetze (1) und (1')
- Assoziativgesetze (2) und (2')
- Distributivgesetze (3) und (3')

um die beiden Gesetze von de Morgan:

(4) $\overline{a \cdot b} = \overline{a} + \overline{b}$ (4') $\overline{a + b} = \overline{a} \cdot \overline{b}$

Aufgabe: Beweise (4) *(Hinweis: Wahrheitstafel!)*

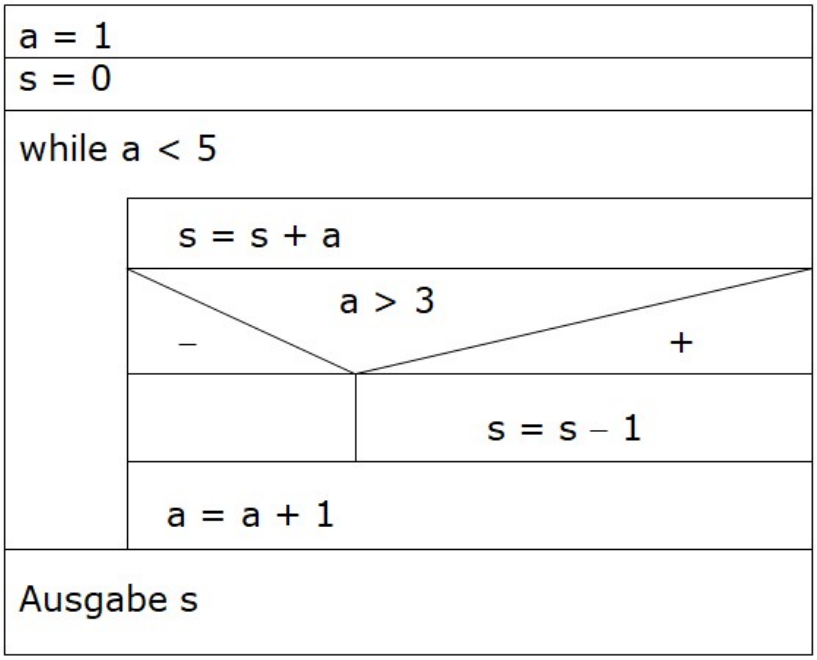
Lösung (Nils):

a	b	a · b	**not (a · b)**	not a	not b	**not a + not b**
0	0	0	**1**	1	1	**1**
0	1	0	**1**	1	0	**1**
1	0	0	**1**	0	1	**1**
1	1	1	**0**	0	0	**0**

Da die Spalten zu **not (a · b)** und **not a + not b** übereinstimmen, gilt:

not (a · b) = not a + not b

16. Erstelle einen in Python geschriebenen Quelltext zu folgendem Struktogramm:



Lösung (Philipp):

```
a = 1
s = 0

while a < 5:
    s = s + a  # oder:  s += a
    if a > 3:
        s = s-1
    a = a + 1

print(s)
```

17. In der Zusammenfassung ***MinSuche_SelectionSort.pdf*** werden die Algorithmen **MinSuche, MinSuche2** und **SelectionSort** noch einmal erläutert.

    Aufgabe: Formuliere bei

    a) MinSuche die for-Schleife zur Bestimmung des kleinsten Elements,

    b) SelectionSort die beiden for-Schleifen zum Sortieren

    jeweils als while-Schleifen und überprüfe die so erhaltenen Python-Programme anhand von Testläufen.

    Lösung zu a) (Benedikt):

```
n = int(input('Anzahl der Datenelemente = '))

a = list(range(1,n+1))

from random import randint
for i in range(0,n):
    a[i] = randint(1,1000)

print('Quelliste:')
print(a)

# Bestimmung des kleinsten Elements
min = a[0]
i = 1
while i < n:
    if a[i] < min:
        min = a[i]
        a[i] = a[0]
        a[0] = min
    i = i + 1

# Ausgabe
print('verarbeitete Liste:')
print(a)
```

Lösung zu b) (Christian):

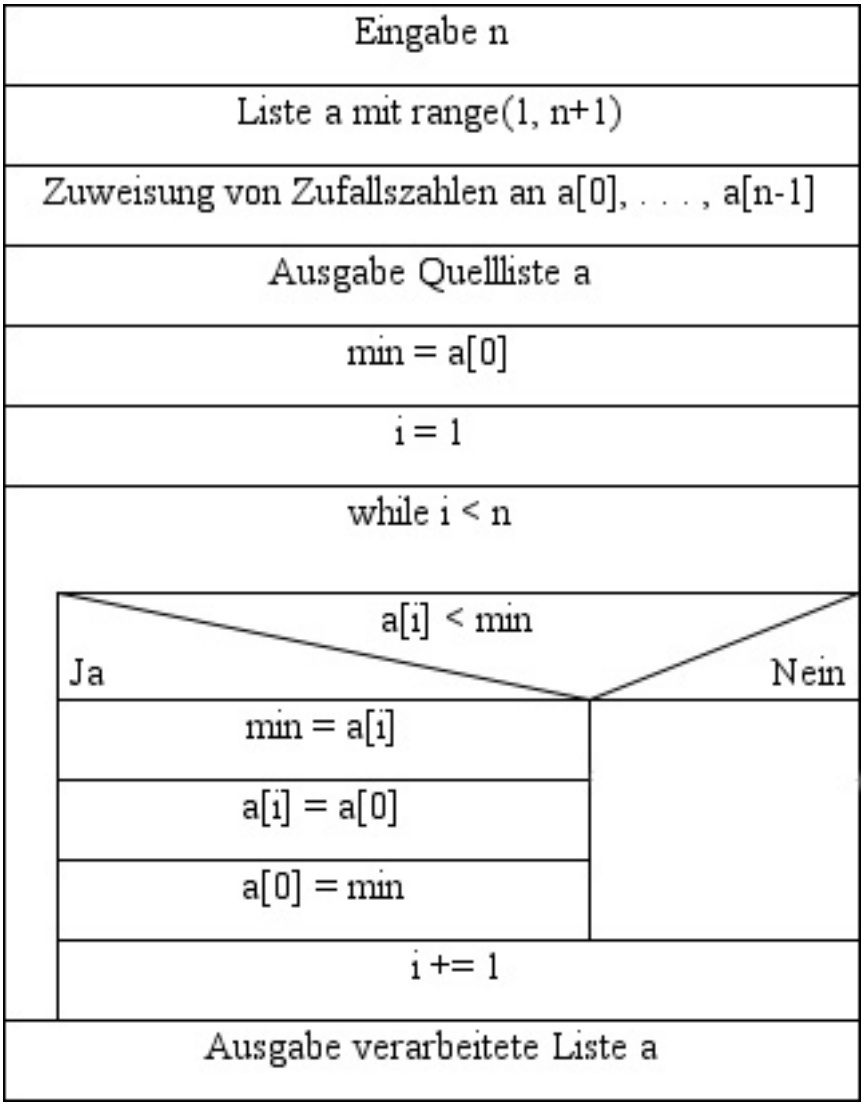
```
n = int(input('Anzahl der Datenelemente = '))

a = list(range(1,n+1))

from random import randint
for i in range(0,n):
    a[i] = randint(1,1000)

# Ausgabe der Quelliste
print('Quelliste:')
print(a)

# Sortieren
j = 0
while j < n - 1:
    i = j + 1
    min = a[j]
    while i < n:
        if a[i] < min:
            min = a[i]
            a[i] = a[j]
            a[j] = min
        i = i + 1
    j = j + 1

# Ausgabe
print('verarbeitete Liste:')
print(a)
```

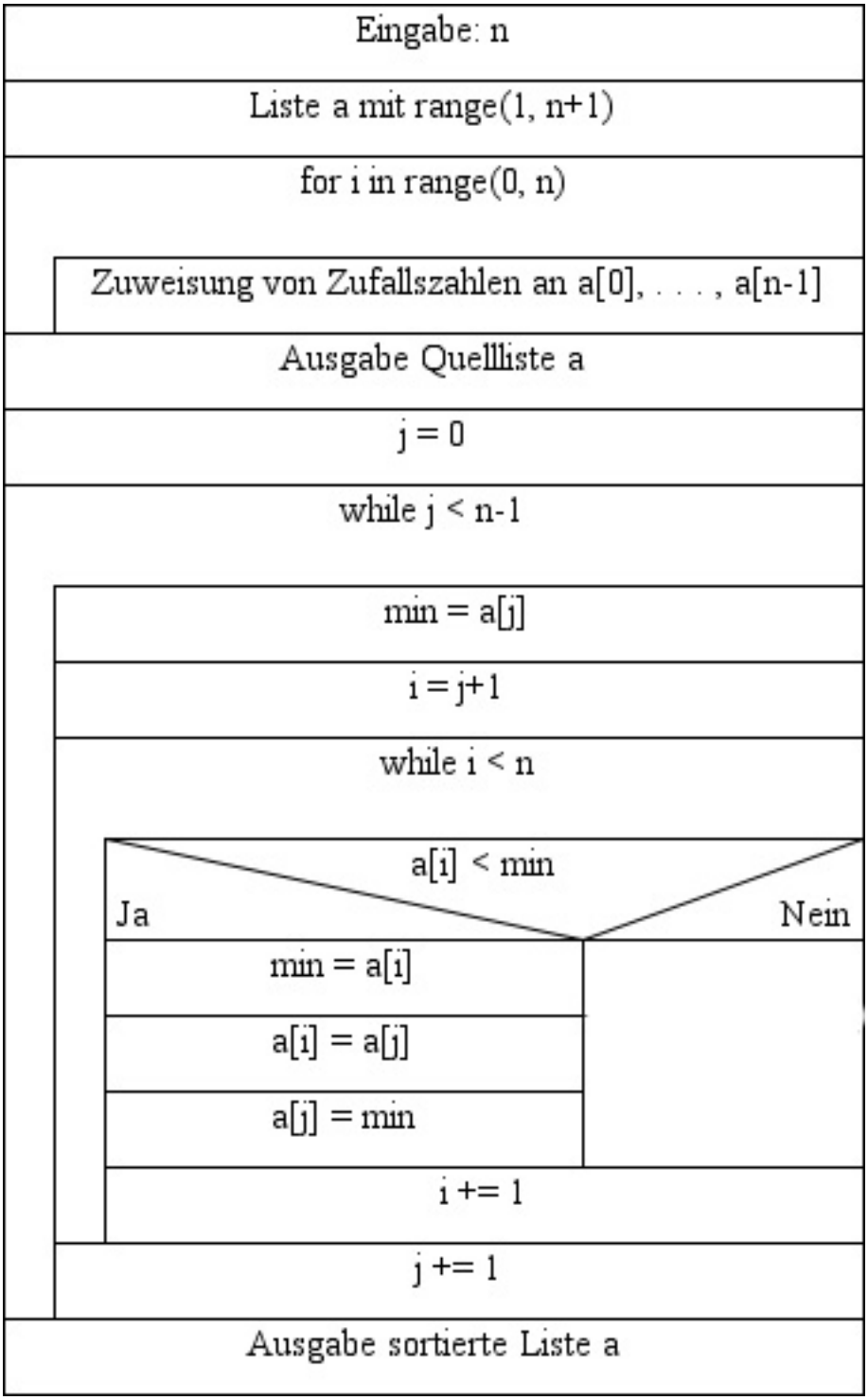
Testlauf (n = 20):

```
Anzahl der Datenelemente = 20
Quelliste:
[726, 236, 827, 926, 294, 774, 431, 133, 346, 434, 480, 167, 310, 690,
841, 344, 434, 411, 697, 301]
verarbeitete Liste:
[133, 167, 236, 294, 301, 310, 344, 346, 411, 431, 434, 434, 480, 690,
697, 726, 774, 827, 841, 926]
```

18. Erstelle jeweils ein Struktogramm zu **MinSuche** und **SelectionSort** (Hinweis: verwende bevorzugt die Versionen mit while-Schleife, Aufgabe 17; für die Zuweisung von Zufallszahlen an die Komponenten des arrays a genügt es zu schreiben: „Zuweisung von Zufallszahlen an a[0], . . . , a[n-1]“)

Struktogramme (Marvin):





19. Freiwillige Aufgabe:
Der Algorithmus SelectionSort (Quelltext: SelectionSort_04-03-2021.py) vertauscht die Inhalte der Speicherplätze a[j] und a[i], j < i < n, immer dann, wenn ein kleineres a[i] als a[j] gefunden wurde; hier gibt es noch Optimierungspotential, um die Rechenzeit insgesamt zu verkürzen. Ergreife diese Möglichkeit und teste! (Die Laufzeit zum Sortieren läßt sich für große Werte von n etwa halbieren.)